

DII.3200.NT40.RG-1

**Defense Information Infrastructure (DII)
Common Operating Environment (COE)**

Version 3.2.0.0

**Application Programmer Interface (API) Reference Guide
(Windows NT 4.0)**

June 13, 1997

Prepared for:

Defense Information Systems Agency

Prepared by:

**Inter-National Research Institute (INRI)
12200 Sunrise Valley Drive, Suite 300
Reston, Virginia 20191**

Table of Contents

Preface	1
1. Introduction	3
1.1 Overview	3
1.2 Referenced Documents	3
2. Calling the DII COE Tools Using the API Toolkit	5
2.1 COEAskUser	7
2.2 COEFindSeg	9
2.3 COEInstError	12
2.4 COEMsg	14
2.5 COEPrompt	16
2.6 COEPromptPasswd	18
Appendix A - Notes	21

This page intentionally left blank.

Preface

The following conventions have been used in this document:

[HELVETICA FONT]	Used to indicate keys to be pressed. For example, press [RETURN].
Courier Font	Used to indicate entries to be typed at the keyboard, operating system commands, titles of windows and dialog boxes, file and directory names, and screen text. For example, execute the following command: A:\setup.exe
<i>Italics</i>	Used for emphasis.

This page intentionally left blank.

1. Introduction

1.1 Overview

This document provides information and guidance needed for using Defense Information Infrastructure (DII) Common Operating Environment (COE) Version 3.2.0.0 public Application Programmer Interfaces (APIs) for Windows NT 4.0. This document contains manual pages for all of the public APIs for the DII COE toolkit for Windows NT 4.0.

This guide is divided into the following section and appendix:

Section	Page
Calling the DII COE Tools Using the API Toolkit Includes manual pages for the following DII COE tools: COEAskUser COEInstError COEPrompt COEFindSeg COEMsg COEPromptPasswd	5
Notes Provides additional information about the COEPromptPasswd tool.	21

Each manual page includes a synopsis, parameters, a description, return values, notes, a reference to related functions, and an example.

Descriptions assume familiarity with the C programming language and with the DII COE development environment.

Reference the *DII COE Integration and Runtime Specification* and the *DII COE Programming Guide (Windows NT 4.0)* for more information about DII COE toolkit.

1.2 Referenced Documents

The following documents are referenced in this guide:

- C DII COE I&RTS:Rev 3.0, *Defense Information Infrastructure (DII) Common Operating Environment (COE) Integration and Runtime Specification Version 3.0*, January 1997
- C DII.3200.NT40.PG-1, *Defense Information Infrastructure (DII) Common Operating Environment (COE) Version 3.2.0.0 Programming Guide (Windows NT 4.0)*, June 13, 1997.

This page intentionally left blank.

2. Calling the DII COE Tools Using the API Toolkit

The API toolkit provides developers with an interface to the DII COE runtime tools. A program can link with a public API to display and retrieve segment information. This section defines the interface for each call, including return values and necessary parameters. This section provides information about using the following APIs:

```
C COEAskUser
C COEFindSeg
C COEInstError
C COEMsg
C COEPrompt
C COEPromptPasswd.
```

NOTE: Reference Appendix A, <i>Notes</i> , for more information about using <code>COEPromptPasswd</code> .

The format of each manual page is as follows:

NAME

Function Name—Provides a brief description of the function.

SYNOPSIS

Presents the calling syntax for the routine, including the declarations of the arguments and the return type. For example:

```
returntype XFunctionName (type1 *arg1, type2 *arg2, type3 *arg3);
```

PARAMETERS

Describes each of the parameters used by the function.

DESCRIPTION

Describes what the function does and what events or side effects it causes.

RETURNS

Describes what the function returns.

NOTE

Provides any notes about the function.

SEE ALSO

Provides a reference to related functions.

EXAMPLE

Provides an example of how to use the function.

2.1 COEAskUser

NAME

COEAskUser—COEAskUser() displays a window with a question and two answer buttons.

SYNOPSIS

```
#include <stdlib.h>
#include <stdio.h>
#include <DIITools.h>

int COEAskUser
(
    char *question,
    char *b1_label,
    char *b2_label
);
```

PARAMETERS

char *question
question - Null terminated string. The question to present to the user.

char *b1_label
b1_label - Null terminated string. The equivalent of the `Yes` button; selection causes the function to return a 1 (`TRUE`).

char *b2_label
b2_label - Null terminated string. The equivalent of the `No` button; selection causes the function to return a 0 (`FALSE`).

DESCRIPTION

The COEAskUser library function creates an interface that prompts the user for a question and gives a choice of `Yes` or `No` buttons to select, where `Yes` and `No` are the default labels assigned. The message question and the answer button labels can be assigned by the user. Null values for `b1_label` and `b2_label` will display default button labels.

RETURNS

COEFAILURE

Failure - The interface could not be displayed. On returning `COEFAILURE`, `COEerrno` is set to:

COEERR_NO_DISPLAY - Window could not be displayed.

COEERR_NO_MESSAGE - Question string not passed in.

TRUE

True - The equivalent `Yes` button selected by the user.

FALSE

False - The equivalent `No` button selected by the user.

NOTE

To support multi-line questions, place `\n` in the desired locations in the question string.

SEE ALSO

COEMsg, COEPrompt, and COEPromptPasswd.

EXAMPLE: COEAskUser

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEAskUser_example COEAskUser_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/include/X11R5 -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5 -L/usr/lib/Motif1.2 -lXm -lXt
-lX11

SOLARIS:
cc -o COEAskUser_example COEAskUser_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm -
lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COEUserPrompts.lib during compilation. (The COEUserPrompts.dll
will be required during execution.)
*/

#include <stdlib.h>
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEAskUser_example */
*****/
int main(int argc, char *argv[])
{
    char          b1_lab[] = "MY_YES";
    char          b2_lab[] = "MY_NO";
    char          message[]="This is my test Message";
    int           ret_val;

    /* Call DII/COE Library Function */
    ret_val = COEAskUser(message, b1_lab, b2_lab);

    return(ret_val);
}

```

2.2 COEFindSeg

NAME

COEFindSeg—COEFindSeg() returns information about a requested segment.

SYNOPSIS

```
#include <stdio.h>
#include <DIITools.h>

int COEFindSeg
(
    int use_installed,
    char *in_segdir,
    char *in_segname,
    char *prefix,
    char *seg_type,
    char *seg_attrib,
    char *actual_dir
);
```

PARAMETERS

```
int use_installed
    use_installed - Installed segments only - 1, any segments - 0. Note: This flag is ignored
    on Windows NT.
char *in_segdir
    in_segdir - Null terminated string. The full directory path leading up to and including
    segdir.
char *in_segname
    in_segname - Null terminated string. The segment name of the segment residing in segdir.
char *prefix
    prefix - Segment prefix returned.
char *seg_type
    seg_type - Segment type returned.
char *seg_attrib
    seg_attrib - Segment attribute returned if available.
char *actual_dir
    actual_dir - Segments actual directory returned.
```

DESCRIPTION

The COEFindSeg library function searches a given segment directory for a given segment name to see if it is on disk and can be read. If no directory is specified, COEFindSeg searches for installed segments by default. When the segment is found, COEFindSeg returns the segment directory, name, prefix, type, and attribute (if the segment has an attribute).

RETURNS

COESUCCESS

Success - The function was successful in finding the requested segment.

COEFAILURE

Failure - The function was not successful in finding the requested segment. On returning

COEFAILURE, COEerrno is set to:

COEERR_SEG_NOT_FND - Requested segment not found.

COEERR_NULL_PARAMS - Both segment name and directory parameters were NULL or empty.

COEERR_FOUND_NOT_INSTALLED - Requested segment was not installed as requested.

NOTE

None.

SEE ALSO

COEAskUser, COEInstError, and COEMsg.

EXAMPLE: COEFindSeg

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEFindSeg_example COEFindSeg_example.c
-I/h/DII_DEV/include -I/usr/include/Motif1.2 -I/usr/include/X11R5
-L/h/DII_DEV/libs -lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5
-L/usr/lib/Motif1.2 -lXm -lXt -lX11

SOLARIS:
cc -o COEFindSeg_example COEFindSeg_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm -
lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COECom.lib, COESeg.lib and COETools.lib during compilation.

*/
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEFindSeg_example */
*****/
int main(int argc, char *argv[])
{
    char segdir[] = "";
    char segname[] = "X Windows";
    int ret_val;
    char out_prefix[8];
    char out_segtype[81];
    char out_segattr[81];
    char out_actualdir[257];

    /* Call DII/COE Library Function */
    ret_val =COEFindSeg(1,segdir,segname,out_prefix,out_segtype,out_segattr,
out_actualdir );
    if( ret_val == COESUCCESS )
    {
        printf("Found segment '%s' prefix '%s' type '%s' at
'%s'\n",segname,out_prefix,out_segtype,out_actualdir);
    }
    else
    {
        printf("Segment '%s' not found \n", segname);
    }

    return(ret_val);
}

```

2.3 COEInstError**NAME**

COEInstError—COEInstError() displays a window with the error message.

SYNOPSIS

```
#include <stdio.h>
#include <DIITools.h>

int COEInstError
(
    char *message
);
```

PARAMETERS

char *message

message - Null terminated string. Error message to display.

DESCRIPTION

The COEInstError library function creates an interface that displays an installation error message. The message can be assigned by the caller of the function. COEFAILURE is always returned. It is up to the calling application to perform appropriate cleanup and return a failed exit status to its parent program (if applicable).

RETURNS

COEFAILURE

Failure - Always returned to signal the calling process that an error has occurred. On returning COEFAILURE, COEerrno is set to:

COEERR_NO_DISPLAY - Window could not be displayed.

COEERR_NO_MESSAGE - Message string not passed in.

NOTE

To support multi-line questions, place \n in the desired locations in the message string.

SEE ALSO

COEMsg, COEPrompt, and COEPromptPasswd.

EXAMPLE: COEInstError

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEInstError_example COEInstError_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/include/X11R5 -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5 -L/usr/lib/Motif1.2 -lXm -lXt
-lX11

SOLARIS:
cc -o COEInstError_example COEInstError_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm -
lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COEUserPrompts.lib during compilation. (The COEUserPrompts.dll
will be required during execution.)
*/
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEInstError_example */
*****/
int main(int argc, char *argv[])
{
    char        message[]="Cannot Access ABC Server";
    int         ret_val;

    /* Call DII/COE Library Function */
    ret_val = COEInstError(message);

    return(ret_val);
}

```

2.4 COEMsg

NAME

COEMsg—COEMsg() displays a window with a message of choice.

SYNOPSIS

```
#include <stdio.h>
#include <DIITools.h>

int COEMsg
(
    char *message
);
```

PARAMETERS

char *message
message - Null terminated string. Message to display to the user.

DESCRIPTION

The COEMsg library function creates an interface that displays a message. The message can be assigned by the caller of the function, which makes it versatile enough to be used as a message for any program.

RETURNS

COESUCCESS

Success - The function was successful in displaying the message.

COEFAILURE

Failure - The function was not successful in displaying the message. On returning

COEFAILURE, COEerrno is set to:

COEERR_NO_DISPLAY - Window could not be displayed.

COEERR_NO_MESSAGE - Question string not passed in.

NOTE

To support multi-line questions, place \n in the desired locations in the message string.

SEE ALSO

COEAskUser, COEInstError, COEPrompt, and COEPromptPasswd.

EXAMPLE: COEMsg

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEMsg_example COEMsg_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/include/X11R5 -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5 -L/usr/lib/Motif1.2 -lXm -lXt
-lX11

SOLARIS:
cc -o COEMsg_example COEMsg_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm -
lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COEUserPrompts.lib during compilation. (The COEUserPrompts.dll
will be required during execution.)
*/
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEMsg_example */
*****/
int main(int argc, char *argv[])
{
    char          message[]="XYZ Has Been Updated";
    int           ret_val;

    /* Call DII/COE Library Function */
    ret_val = COEMsg(message);

    return(ret_val);
}

```

2.5 COEPrompt

NAME

COEPrompt—COEPrompt() displays a window with a user-assigned prompt and an editable text area for user input.

SYNOPSIS

```
#include <stdio.h>
#include <DIITools.h>

int COEPrompt
(
    char *prompt,
    int max_len,
    char *text_return
);
```

PARAMETERS

char *prompt
 prompt - Null terminated string. Prompt text to display.

int max_len
 max_len - Maximum length of text accepted in the editable text field.

char *text_return
 text_return - Null terminated string. Text returned—must be allocated to max_len + 1.

DESCRIPTION

The COEPrompt library function creates an interface that displays a prompt and a text area to accept user input. The prompt can be assigned by the caller of the function. The user input will be sent to stdout.

RETURNS

COESUCCESS

 Success - The function was successful in displaying the prompt and the text area.

COEFAILURE

 Failure - The function was not successful in displaying the prompt and the text area. On returning COEFAILURE, COErrno is set to:

 COERR_NO_DISPLAY - Window could not be displayed.

 COERR_NO_MESSAGE - Prompt string not passed in.

NOTE

To support multi-line questions, place \n in the desired locations in the message string.

SEE ALSO

COEAskUser, COEInstError, COEMsg, and COEPromptPasswd.

EXAMPLE: COEPrompt

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEPrompt_example COEPrompt_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/include/X11R5 -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5 -L/usr/lib/Motif1.2 -lXm -lXt
-lX11

SOLARIS:
cc -o COEPrompt_example COEPrompt_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm
-lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COEUserPrompts.lib during compilation. (The COEUserPrompts.dll
will be required during execution.)
*/
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEPrompt_example */
*****/
int main(int argc, char *argv[])
{
    char        message[]="Please Enter What Server To Access : ";
    int         input_text_len = 15;
    int         ret_val;
    char        ret_text[16];

    /* Call DII/COE Library Function */
    ret_val = COEPrompt(message, input_text_len, ret_text);

    return(ret_val);
}

```

2.6 COEPromptPasswd

NAME

COEPromptPasswd—COEPromptPasswd() displays a window with an optional `Password` prompt message and an optional `Verify` prompt.

SYNOPSIS

```
#include <stdio.h>
#include <DIITools.h>

int COEPromptPasswd
(
    int max_text_len,
    int user_wants_verify_prompt,
    char *prompt,
    char *passwd_return
);
```

PARAMETERS

```
int max_text_len
    max_text_len - Maximum characters the user may enter for the password.
int user_wants_verify_prompt
    user_wants_verify_prompt - Flag whether to display a Verify prompt.
char *prompt
    prompt - Null terminated string. Optional prompt to display text.
char *passwd_return
    passwd_return - Password returned.
```

DESCRIPTION

The COEPromptPasswd library function creates an interface that displays an optional `Password` prompt message and an optional `Verify` prompt. The optional `Verify` prompt can either be (1) displayed by passing in `TRUE` or (2) not displayed by passing in `FALSE` through the third parameter. The password entered will be sent to `stdout`.

RETURNS

COESUCCESS

Success - The function was successful. (This function is always successful if the `Verify` prompt is not displayed.)

COEFAILURE

Failure - The interface cannot be displayed, or the `Password` and `Verify` text are not the same. On returning `COEFAILURE`, `COEerrno` is set to:

```
COEERR_NO_DISPLAY - Window could not be displayed.
COEERR_EMPTY_FIELD - No Password text or Verify text returned.
COEERR_NO_MATCH - Password and Verify string do not match.
```

NOTE

The password has a default maximum length of 40 characters when `max_text_len` is less than or equal to zero. The maximum password length can be increased or decreased by the caller of the function via the `max_text_len` field. If the input maximum length is exceeded, the input will be truncated.

To support multi-line questions, place `\n` in the desired locations in the message string.

SEE ALSO

`COEAskUser`, `COEInstError`, `COEMsg`, and `COEPrompt`.

EXAMPLE: COEPromptPasswd

```

/*
To build this routine use the following command (substitute your
location for the DII_DEV directory, Motif libraries and includes) :

HP:
cc -Aa -o COEPromptPasswd_example COEPromptPasswd_example.c
-I/h/DII_DEV/include -I/usr/include/Motif1.2 -I/usr/include/X11R5
-L/h/DII_DEV/libs -lCOETools -lCOE -lPrintClient -L/usr/lib/X11R5
-L/usr/lib/Motif1.2 -lXm -lXt -lX11

SOLARIS:
cc -o COEPromptPasswd_example COEPromptPasswd_example.c -I/h/DII_DEV/include
-I/usr/include/Motif1.2 -I/usr/openwin/include -L/h/DII_DEV/libs
-lCOETools -lCOE -lPrintClient -L/usr/openwin/lib -L/usr/lib/Motif1.2 -lXm
-lXt -lX11 -lgen

NT:
In your compile environment, make sure your include file path
includes DII_DEV\include and your library path includes DII_DEV\libs.
Link COEUserPrompts.lib during compilation. (The COEUserPrompts.dll
will be required during execution.)
*/
#include <stdio.h>
#include <DIITools.h>

/*****
/* COEPromptPasswd_example */
*****/
int main(int argc, char *argv[])
{
    int          input_text_len = 10;
    int          ret_val;
    char         ret_passwd[11];

    /* Call DII/COE Library Function */
    ret_val = COEPromptPasswd(input_text_len, 1,
        "Please enter passwd for MyUser", ret_passwd);
    if (ret_val == COESUCCESS) {
        printf("Password Is Correct\n");
    }
    else {
        printf("Password Is Incorrect\n");
    }

    return(ret_val);
}

```

Appendix A - Notes

COEPromptPasswd

The password has a default maximum length of 40 characters when `max_text_len` is less than or equal to zero. The maximum password length can be increased or decreased by the caller of the function via the `max_text_len` field. If the input maximum length is exceeded, the input will be truncated.

The software does not check if the `max_text_len` of the password is less than or equal to zero. Therefore, the value will not default to 40.

This page intentionally left blank.